Apellidos: **Nombre:** **Matrícula:**

Examen

105000119 - Programación para Sistemas

10MI-Grado en Matemáticas e Informática

Lenguajes y Sistemas Informáticos e Ingeniería de Software
ETSI Informáticos
Universidad Politécnica de Madrid

Curso 2017/2018 - Enero 2018

Normas

- El examen puntúa sobre **18 puntos**.
- La duración total del mismo es de **65 minutos**.
- Se deberá tener el DNI o el carnet de la UPM en lugar visible.
- No olvidar rellenar **apellidos, nombre y número de matrícula** en cada hoja.
- La solución al examen se proporcionará antes de la revisión.

Cuestionario

(1 punto) 1. Sea la siguiente instrucción en lenguaje Bash:

```
cd /tmp && mkdir MiDir
```

Impleméntela de manera equivalente con un condicional.

Solución:

```
if cd /tmp; then
  mkdir MiDir
fi
```

(1 punto) 2. Sea el fichero *miscript.sh* :

```
#! /bin/bash
echo $#
echo $0
echo $1
echo $2
```

Indique la salida que corresponde a la llamada:

```
./miscript.sh uno dos
```

Nota: suponga que el programa 'bash' se encuentra en '/bin/bash', y que el fichero 'miscript.sh' tiene permiso de ejecución.

Solución:
2
./miscript.sh
uno
dos

(1 punto) 3. Sea el fichero *conf* con permiso de lectura que contiene la instrucción:

MIVAR=10

Indique la salida que se mostrará al ejecutar en un intérprete Bash las siguientes instrucciones:

```
MIVAR=5
source conf
echo $MIVAR
```

Solución: 10

(1 punto) 4. En Unix, indique la instrucción correspondiente para establecer el permiso de ejecución de un fichero *miscript.sh*.

Solución: chmod +x miscript.sh

(1 punto) 5. En Bash, indique cómo lanzar el comando *ls *.txt* de forma que (a) la salida estándar se redirija al fichero *salida.txt*, y que (b) el error estándar se redirija al fichero *error.txt*. (Deberá haber sólo una llamada a *ls*.)

Solución:

```
ls *.txt > salida.txt 2> error.txt
```

(1 punto) 6. En un script Bash, indique las instrucciones para, en el caso de que no se hayan pasado exactamente 2 argumentos, se acabe el script con resultado de status 1.

REQUISITO: se tiene que emplear necesariamente una instrucción condicional if-then.

Solución:

```
if test $# -ne 2
then
  exit 1
fi
```

(1 punto) 7. En lenguaje C, indique la instrucción que muestra en salida error el texto:
Mensaje de error

Solución:

```
fprintf(stderr, "Mensaje␣de␣error");
```

(1 punto) 8. Sea el siguiente extracto de una función 'mifunc':

```
int mifunc(int n, float x) {
  /* codigo de la funcion */
}
```

Indique el prototipo (cabecera) de la función.

Solución:

```
int mifunc(int n, float x);
```

También

```
int mifunc(int, float);
```

(1 punto) 9. Dado el siguiente código,

```
#include <stdio.h>
int main (int argc, char*argv[])
{
int minimo=10, maximo=15;
while(minimo<=maximo) {
  maximo=maximo--;
}
printf("VALOR␣%d␣\n", maximo);
}
```

indicar la salida de su ejecución.

Solución: 9

(1 punto) 10. Dada la siguiente estructura,

```
struct alumno
{
  char nombre[50];
  char apellidos[80];
  int dni;
} nuevo ;
```

Cuál de las siguientes opciones es la correcta para inicializar el campo nombre?

a) nuevo.nombre=”Roberto”; b) nuevo->nombre=”Roberto”; c) strcpy(nuevo.nombre,”Roberto”); d) Ninguna de las anteriores

Solución: c)

(1 punto) 11. Dado el siguiente código,

```
int main (int argc, char*argv[])
{
  int A=1, B=2, C=3, *P1, *P2;
  P1=&A;
  P2=&C;
  *P1=(*P2)+1;
  printf("VALOR␣DE␣A␣%d␣B␣%d␣C␣%d␣\n",A,B,C);
}
```

indicar la salida de su ejecución:

Solución: VALOR DE A 4 B 2 C 3

(1 punto) 12. Dado el siguiente código,

```
int main (int argc, char*argv[])
{
int A[] = {12, 23, 34, 45, 56, 67, 78, 89, 90};
int *P, resultado;
P=A;
resultado=*P+2;
printf ("RESULTADO␣1␣%d␣\n" , resultado);
resultado=*(P+2);
printf ("RESULTADO␣2␣%d␣\n" , resultado);
}
```

indicar la salida de su ejecución.

Solución: RESULTADO 1 14
RESULTADO 2 34

(1 punto) 13. Dado el siguiente código

```
int main (int argc, char*argv[])
{
int i,j=3;
for (i=0;i<3;i++) {
 switch(j+2) {
```

```
  case 6: j=j+2;
  case 5: j=j-1;break;
  case 4: j=j+2;break;
 }
}
printf ("VALOR␣%d␣\n", j);
}
```

indicar la salida de su ejecución.

Solución: VALOR 5

(1 punto) 14. Dado el siguiente código,

```
int main (int argc, char*argv[])
{
int x=1, y=2, *ip;
ip=&x;
*ip=*ip+1;
y=*ip+1;
printf("X␣%d␣Y␣%d␣IP␣%d␣\n", x,y,*ip);
}
```

indicar la salida de su ejecución:

Solución: X 2 Y 3 IP 2

(1 punto) 15. Indique la línea necesaria para reservar memoria dinámica para un vector de n números enteros.

Solución:

```
v=(int*)malloc(sizeof(int)*n);
```

(1 punto) 16. Dado el siguiente código

```
int main (int argc, char*argv[])
{
int i;
for (i=0; i<argc; i++)
printf(" %s\n",argv[i]);
}
```

si una vez compilado se ejecuta de la siguiente manera:

./programa HOLA AMIGO

indicar su salida

Solución: ./programa
HOLA
AMIGO

(1 punto) 17. Si un programa en C, está compuesto por dos ficheros fuentes main.c y suma.c y se quiere crear un ejecutable llamado *calculo*, indicar como habría que compilar el programa.

Solución: gcc -o calculo main.c suma.c

(1 punto) 18. Se va a utilizar la siguiente declaración de doble puntero en lenguaje C:

```
char **ppchar;
```

la cual creará una variable tipo puntero:

```
ppchar
-------
|     |-->
-------
```

Se desea llegar a obtener el siguiente diagrama:

```
ppchar
-------   -------   -----
|     |-->|     |-->|'A'|
-------   -------   -----
```

donde ppchar apunta a un puntero que a su vez apunta a un carácter al que se ha asignado el valor 'A'.

Complete el código siguiente para conseguirlo, debiéndose después liberar la memoria dinámica que se haya asignado antes de finalizar el programa.

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {

  char **ppchar;
  /* Inicio del c\'odigo a completar */


  /* Fin del c\'odigo a completar */
  return 0;
}
```

Solución:

```
#include <stdio.h>
#include <stdlib.h>

int main( void ) {

  char **ppchar;

  ppchar = ( char ** ) malloc( sizeof( char * ) );
  if ( ppchar == NULL) { exit( 1 ); }

  *ppchar = ( char * ) malloc( sizeof( char ) );
  if ( *ppchar == NULL) { exit( 1 ); }

  **ppchar = 'A';

  /* liberar la memoria din\'amica asignada */
  free( *ppchar );
  free( ppchar );

  return 0;
}
```